Amendments to the Claims:

This listing of claims will replace all prior versions, and listing, of claims in the application:

Listing of Claims:

Claims 1-20 (canceled)

Claim 21 (new): A method of protecting memory locations associated with an embedded system, the method comprising:

starting a state machine having at least a change state and a normal state;

upon starting the state machine, entering the change state when an indication is present that data needs to be persisted to protected memory locations, otherwise entering the normal state;

in both the change state and the normal state, starting a write filter that intercepts writes to the protected memory locations and stores the writes in a write cache;

in the normal state, i) identifying requests for critical writes to the protected memory locations, ii) describing the critical writes in at least one update file stored in unprotected memory locations, but not persisting the critical writes to the protected memory locations, and iii) setting the indication that data needs to be persisted to the protected memory locations;

in the change state, causing the critical writes described in the at least one update file to be persisted to the protected memory locations, and initiating a reboot of the embedded system; and

before re-entering the change state or the normal state, rebooting the embedded system.

Claim 22 (new): The method of claim 21, further comprising:

emptying the write cache upon reboot of the embedded system.

Claim 23 (new): The method of claim 21, further comprising:

running applications of the embedded system in the normal state; and

not running the applications of the embedded system in the change state.

Claim 24 (new): The method of claim 21, wherein the step of causing the critical writes described in the at least one update file to be persisted to the protected memory locations comprises:

clearing the write cache;

after clearing the write cache, applying the critical writes described in the at least one update file to the write filter, thereby causing the critical writes to be stored in the write cache; and then

issuing a command to persist the critical writes stored in the write cache to the protected memory locations.

Claim 25 (new): The method of claim 24, further comprising, after successfully applying the critical writes described in the at least one update file to the write filter:

deleting the at least one update file; and

deleting the indication that data needs to be persisted to the protected memory locations.

Claim 26 (new): The method of claim 21, wherein the step of setting the indication that data needs to be persisted to the protected memory locations comprises:

writing a file name of each update file to a data file.

Claim 27 (new): The method of claim 21, wherein each update file is assigned a file name based on a time stamp.

Claim 28 (new): The method of claim 21, further comprising:

in the change state, if an update executable exists executing said update executable.

Claim 29 (new): The method of claim 28, further comprising:

putting the state machine in a sleep mode during the execution of the update executable.

Claim 30 (new): An embedded system, comprising:

a processing unit responsive to an operating system for executing applications to perform functions of the embedded system;

a main memory location storing the operating system of the embedded system, said operating system providing a write filter that protects the operating system from writes;

a secondary memory location for storing software and data; and

a state machine that executes automatically upon boot of the embedded system and causes the embedded system to operate in either a normal state or a change state, wherein:

during operation in the normal state, i) the applications are run, and ii) when a critical write to the operating system is requested, the critical write is not persisted to the main memory location and an update file is generated in the secondary memory location, wherein the update file stores the critical write until the embedded system enters the change state; and

during operation in the change state, no applications are run and the update file is used to update and persist the critical write to the operating system;

wherein the operating system is configured to invoke the write filter in both the normal state and the change state.

Claim 31 (new): An embedded system, comprising:

a state machine that, in conjunction with booting of the embedded system, assumes one of two states, the two states being, a normal state in which applications are executed, in which a write filter intercepts writes to protected memory locations and redirects them to unprotected memory locations, in which the writes to the protected memory locations are not persisted to the protected memory locations, and in which critical writes to the protected memory locations are identified and described in at least one update file located in the unprotected memory locations; and

a change state, entered across a boot from the normal state, in which the write filter is cleared and critical writes described in the at least one update file are then applied to the write filter, in which the write filter causes the critical writes to be stored in a write cache, and in which the critical writes stored in the write cache are subsequently persisted to the protected memory locations.

Claim 32 (new): The embedded system of claim 31, wherein the applications are not run during the change state.

Claim 33 (new): The embedded system of claim 31, wherein the critical writes include writes to a system registry.

Claim 34 (new): The embedded system of claim 31, wherein, during the change state, the critical writes update an operating system of the embedded system.

Claim 35 (new): The embedded system of claim 31, wherein the protected memory locations store an operating system of the embedded system.